# BEEBLET

Send S.A.E. for list of DEALS
& DISCOUNTS available to MEMBERS

ISSN 0112-3351

## CONTENTS

President's Report 2
Editorial Notes 3
For Sale 3
Regional News 4
HELP 5
Hints 7
The Old Switcheroo 8
Ancient Greek Characters 10
Beginners and Listings 13
Windows 15
Macros in VIEW 22
Bright Ideas 24
Variables and Procedures 25
Decay Series 26
Review - Oxford Pascal 27
String Handling Package 30
Top of the Pops 33
..SAS. 36
Thoughts on Multi Player Games 38
Hall of Fame 39
Local Groups 40

VOLUME 3 NUMBER 6 JULY 1985

**BBC/ACORN**

COMPUTER USER GROUP
NZ (inc)

# PRESIDENT'S REPORT - JULY 1985

At last it's official - after a year of bureaucratic dealings we are now formally registered under our new name of the BBC/Acorn Computer User Group N.Z. (Inc). In case you've forgotten, the name change is in recognition of the growing numbers of Acorn Electron owners in the Group, and to allow for other Acorn computers which won't carry the BBC name.

Due, I'm sure, in no small part to my criticism of the pricing of the BBC B+ in last months "Beeblet", I understand that the price in Britain has been reduced by 50 pounds. Acorn must also be feeling the competitive pressure from the higher end of the market - the Atari 520ST (with 512 Kbytes) is being advertised at 575 pounds (with VAT) and the price of the Apricot F1e with 256K and a 320K 3.5 inch drive included has dropped to 620 pounds. To add insult to injury, for 30 pounds you can buy a program called B-Tran which will enable you to run BBC Basic programs on the Apricot. It does it by translation to GW-Basic, converting PROC calls to horrible GOSUBs, etc. Rumour has it that the final debugging of the program was done by Barsons NZ.

A few more details on the B+ - it definitely doesn't use the double density capabilities of the WD disk controller chip, and nor does it provide the ADFS used in the Electron. The additional 32K of memory is mapped from address &3000 to &AFFF - meaning that the top 12K overlaps the sideways ROM area and can possibly be used as sideways RAM, although Acorn will no doubt frown on that. The B+ is not expected to be on sold in NZ until about October - reason being that it's selling so well in Britain (?). Meanwhile, watch out for the model C. This will use the 16 bit version of the 6502 microprocessor, and will give the full BBC graphics resolution of the BBC - 1280 by 1024. Acorn will probably be aiming to have it out in Britain by Christmas.

Membership renewals have been encouragingly high. The Committee have been working on alternative methods of producing the magazine and it looks like we may be able to make some significant savings soon, with a possible reduction in subscriptions. More on that next month. However, the best way of lowering costs (and hence subscriptions) per member is to increase membership. For that we need publicity. A publicity release on the name change will be going out shortly, and we'll be having a stand at the Wellington Computer Show this month. The modem project should also raise our public profile. (Current status - the original prototype now has filtering and on tests has easily passed the Post Office specifications. A PCB prototype will be built by us and submitted for type approval this month. We'll probably get a local firm to build them. With expected production in the hundreds, it's just not worth the hassle of doing it ourselves. The cost to you will still be not much above $150 and certainly below $200.) Meanwhile, you can do your bit. If you have friends with BBC's or Electrons who aren't in the Group, point out all the advantages and encourage them to join.

Neil Gordon

## EDITOR'S NOTES

Congratulations Steve Williams - winner of the early bird membership refund. See what can happen if you contribute regularly to Beeblet, pay your sub. promptly, and have the luck of the devil!

COMING in the next few issues ..
3D text, MEGA (Neville T.), a program merger, misc utilities, a report on the U.G. survey, more short listings and a raft of reviews.... And anything else I can get my hands on...

BEEBLET is edited by Bruce Wills who may sometimes be tracked down on 861-662 (h) or 829-929 x8896 (wk).

### DISCLAIMER

The views of the reviewers, the Editor and other contributors are not necessarily shared by the BBC/ACORN COMPUTER USER GROUP OF NZ INC.

## FOR SALE:

### CASSETTE TAPES

C10s, boxed, leaderless - available from the UG in multiples of 20 : $30-00 plus $2-00 p&p.

### MICRO COVERS

Designed to fit the BBC micro. Attractive opaque reinforced plastic, $5-50 posted. To Iain, Box 26078, Epsom, Auckland 3.

### FLOPPY DISCS

3M Brand 5 1/4" DSDD discs - available from the UG ex-stock in (cardboard) boxes of 10: $55-00, which includes p&p

Merlin Double-Density DFS. Treats both sides of the disk as a single surface, allows a single file of up to 720K, and up to 159 files per disk. It increses speed three times and is 95% Acorn compatible. Price $330-00.

Phone John Colvin, Wellington 662-551.

# REGIONAL NEWS

## AUCKLAND

Steve Williams (he who won the membership refund ballot) reports:

The meeting on 12 June was addressed by Dave Smith of Equity Research who described how he based his Investment advisory business on a package running on a BBC. Dave also had a host of other ideas about how a small business could be run using a microcomputer, which made interesting listening.

Mike's beginner's section is proving very popular, with approx 20 people adjourning to the next room for a problem / advice session, while the main meeting was in progress. Many people are reluctant to ask what they may consider to be "silly" questions in front of a large group of "experts" (who may have forgotten that they too were once novices) and so the idea of having a separate group aimed specifically at the novice is a very good one.

Next meeting is on interface options, but after that, who knows?

*******************************

## HAWKES BAY

Paul Hobbs reports:

We are a lively group with a growing number of Electron users meeting on the second Monday of each month 7.00 - 9.00pm alternating between Napier and Hastings. We are keen to correspond further with other user groups.

Members receive a monthly newsletter, an introductory software package and access to the group's Public Domain software library. Discount is available on the first subscription to the National User Group and on Cassettes, discs etc. The membership fee is currently $14 per annum ($5 for school students.)

Forthcoming meetings will include the following:

July: Monitors & TVs, Sound
August: Electron + add-ons, Graphics
September: Spreadsheets, Structure.

Contact Kendall on 435,624 (membership), Paul on 85,834 (newsletter),Bob on 56,015 (library) or write to P.O. Box 338, Napier enclosing S.A.E.

# HELP

## A forum conducted by Tim Ryan

Hmm, well. Tim was a little tied up this time and did not receive much by way of problems - he nevertheless still provided invaluable assistance. Please keep those problems rolling in - for every person who asks about a problem there are at least half a dozen others pondering... I have collected a few queries however : Don't blame Tim for my answers!

Alan J. congratulates Royden on his draughting work on his lectern, but asks..

Q1. How was the print in the article arranged at the right angles?

A1. Double sided cellotape.. The boxed text was produced with a sketch program written by Tim (this could have been used to do the diagrams, given time). I also have another drawing program called ASTAAD2, published by BEEBUG, which is capable of positioning text of varying sizes at the approriate angles and positions around a diagram. I will try to commission an article - volunteers ?

Q2. Alan also has a problem with his Star Radix 15f printer and Star Printmaster Rom. The *GPRINT command, used to produce special effects text, misses the "top" line of any inverse type - in whichever orientation it is printed. Quad width triple height upside down text misses a stripe in printing - this is apparently related to DIP switch settings (C3). The manual is also rather sparse in examples of available print types.

Can anyone offer Alan advice here? Anyone else having similar problems?

Q3. Stephen P. asks about VIEW reviews, and whether it is necessary to buy a print driver program for VIEW or whether the one in the May 84 Beeblet would work on his Electron, and is importing software direct a good idea in view of high prices locally.

A3. There is a brief comparitive review of VIEW in January 1985 Acorn User which does raise a number of pertinent points for prospective WP purchasers. A print driver program is required with VIEW, however I understand that the latest version has a built in provision for you to produce a customised print driver. I doubt that the Beeblet driver would be suitable for an Electron as it stands because of the different hardware (ie Electron expansion interface). It would not be a particularly good project for an inexperienced user to try and implement. Could an Electron owner please comment ?

Purchasing overseas involves balancing risk and service against cost - we are well served in Wellington by the major BBC dealer, and I and others have greatly appreciated the ability to conveniently return defective products, obtain advice and assistance. I have even returned software which did not meet my requirements. Dealing with a retailer on the other side of the world can have its problems - still, it can be a lot cheaper.

Finally, the word-processor you have sounds similar to WORDWISE which I use to produce the magazine. I do not find 40 column editing (and having to return to the menu to view the text in 80 columns) of great concern : 40 columns on the Beeb (the dreaded Mode 7) uses considerably less memory for the screen and thus has more space for text. Whether the additional cost of VIEW is justifiable depends upon the frequency of use, the sort of documents you generally produce (eg short letters, long dissertations) and how much you hate 40 column editing!

Q4. Brian S. has been suffering problems loading from cassette into his Electron.

A4. As a confirmed cassette hater I have been surprised how reliable the Acorn systems usually are. Such problems usually fall into one of three areas:
-equipment compatibility
-operation
-maintenance

Most cassette recorder / players are designed to reproduce sound, not clean computer signals. Some brands work better than others - cost, or "quality" is not necessarily correlated with performance here. Borrow a few different models - any improvement?

Volume seems to be less critical with Acorn machines than some others, nevertheless its worth doing some systematic experimenting up and down the scale. Check that all connections are secure, and for other nearby electrical appliances which might cause interference.

Clean the playback / recording heads and the plug connections. (Those unfamiliar with the mechanism of tape recorders might find a brief chat with a local retailer or service person quite rewarding.) Use either an appropriate cleaning fluid (audio dealer or chemist) or head cleaning tape. The heads could be magnetised : a demagnetising cassette cartridge can be obtained from audio shops. If constant adjustments to the azimuth angle of the tape is required, perhaps it is constanly working loose - I know of someone who applied a spot of glue once they got the adjustment right... Listening to the signal can sometimes provide clues : I know of one case where a recorder fault was adding a high pitched signal of its own..

I have two cassette recorders. The cheap one loads almost anything (with one or two notable exceptions). The more sophisticated one loads nothing.

## HINT

Alan B. has suggested another cause for the 'NO PROC' error message - when a program has a main line with a list of procedures and/or functions, as in the following example, it will crash if there is an 'END' statement missing from the main line :

```
10  REM start of the program
20  FOR variable=0 to 10
30  PROCedure
40  NEXT variable
100 DEF PROCedure
110 PRINT variable
120 ENDPROC
```

All that is missing is a line

```
50  END
```

## HINTS

Don A. offers a U.D. Key version of the System Integer Chart program in the March Beeblet :

```
*Key8  z%=65: |N:F.f%=&404to&4688.4:If!f%<>0
P.CHR$32CHR$32Chr$z%"%  =  ";!f%"/
&";~!f%:z%=Z%+1:N.EL.z%=z%+1:N.|M|O
```

This switches on the page mode, then lists the values of the resident intgers a% - Z5 in decimal and hex, finally switching page mode off again.

Any value other than zero will disable sound output in *fx210(value). Don has also speed up the cursor movement in VIEW using *fx12,4 (default is *fx12,8).

# THE OLD SWITCHEROO

by Geoff P.

I like mode 3. Even in the bad old days before I bought VIEW it was my favourite mode for typing programs,debugging listings and the like. In fact I had a permanently programmed function key just to switch me back and forth. You can imagine my delight in finding an even more permanent solution.

If you take the lid off your Beeb you'll see at the very front of the keyboard, by the DELETE key, a series of holes, or to be more precise, two parallel lines of eight holes. If you undo the keyboard, (a nut at either end and unplug the speaker and main ribbon cable), and look underneath, you'll see that each o these holes matches a little splodge on the printed circuit board. Even I, an electronic neanderthal, realised that these were two sides of a circuit awaiting reconnection. But to what ends? Read on ... Now the detective work begins. Tucked away in the back of my Watford DFS manual is a small section titled "Links" which refers to those holes in your keyboard. (At this point I should mention that the more affluent - those that could afford a Beeb with a built-in DFS - should have a block of DIP switches in place of holes. This makes life a lot easier.) The main purpose of these switch links, or holes, is to allow the computer to use a wide variety of disc drives and the different timings that they require. The last three however are "high, middle and low bit of graphics mode at BREAK".What that delightfully succinct phrase means is that by making and breaking those last three connections you can change the mode that the computer jumps into every time you hit BREAK. You can start up in mode 3, or any other mode, if you like! Armed with a couple of pieces of wire I discovered the following connections necessary for the desired mode changes:

| MODE DESIRED: | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|---|
| LINK No. | 6 | X | X | X | X | | | | |
| | 7 | X | X | | | X | X | | |
| | 8 | X | | X | | X | | X | |

Everywhere you see an X simply make a connection across the links for the mode you desire. For example, mode 3 requires just a single connection across link 6. (Links are numbered from left to right.) I used a fine piece of wire inserted from the top of the keyboard and soldered on each side. If I ever want to change back I just need to snip the wire. Of course if you're ambitious you could always fit a DIP switch but I don't even know what one is, let alone where to buy one.

PS. Does the Electron have a similar set-up? Perhaps someone might like to ferret this out and report back...

Editor's note:

I commented on implementing this in software (using *FX255) in the last issue. To clarify further, these 8 links set 8 bits (which determine READ/WRITE SET UP OPTIONS) as follows:

| LINK | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| BIT | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |

Closing the link (denoted above with an "x") sets the bit to 0, leaving it open sets it to 1.

Links 6,7, and 8 correspond to bits 2,1 and 0 respectively and the three bit value of these is a binary number which sets the default MODE number:
eg. MODE 0, xxx, set all bits to 0, ie 000 = 0 .
MODE 5, -x-, set bit 1 to 0, ie 101 or $(2^2=4)$ +0+ $(2^0=1)$ =5.

For those a little unsure on binary to decimal conversion, use the following table:

| Bit Position | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Decimal | 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 's (2 to the power of the bit position) |

Link 5, or bit 3, can set up the BREAK key to produce the SHIFT+BREAK disc autoboot.

Links 3 and 4, or bits 4 and 5, set disc drive timings (see p 246, A.U.G.)

Links 1 and 2 are currently unassigned.

*FX255,215, followed by BREAK, sets up the binary number 11010111.
From right to left
111 sets MODE 7
10 set the DD step time to 6, settle time to 16, head load to 0 (Whatever that means).

# ANCIENT GREEK CHARACTERS

Neville T.

The accompanying program should be an interesting, if not fun, diversion. It defines an alternative character set... in Ancient Greek. The Ancient Greek alphabet had 24 characters. You are probably familiar with some of these eg. alpha, beta, gamma, delta, omega, sigma etc...

Once set up, the program prints a table of the entire alphabet, upper and lower case. You then have an option - entering English or Greek text. This is based upon the supposition that each letter in the English alphabet has a corresponding letter in the Ancient Greek alphabet. If you pick the English option, the program takes the string you enter and converts each character you enter into it's Greek equivalent. The Greek option allows you to enter Greek characters directly from the keyboard.

The capital letters of the Greek alphabet in many cases bear a close resemblance to the capitals in our own alphabet, although in different places. You will get the most interesting effects from entering text in lower case.

As a brief description of the program as it stands, FNgreek converts the ASCII value of an English letter into the ASCII value of the user-defined character of the Greek equivalent. Line 350 holds the name of each of the Greek letters. Line 370 is a short procedure to print a double height letter in any mode other than mode 7. It uses OSWORD call with the accumulator set to &A to read the character definition, then define two characters - one of each half of the letter (see the article "Teletext - by any mode" Beeblet August 1984).

```
 10 REM The Greek Alphabet
 20 REM By Neville Tomlinson
 30 REM 16-April-1985
 40 :
 50 *TV255
 60 MODE1
 70 A$="The Greek Alphabet":COLOUR3
 80 PRINTTAB(11,0);
 90 FORZ%=1TOLENA$:PROCdbl(ASCMID$(A$,
     Z%,1)):VDU11
100   NEXT
110 PROCchars
120 COLOUR2:PRINTTAB(0,2);
130 FORX%=65TO76
140   PRINT
150   READA$,B$
160   PRINTTAB(5)CHR$(FNgreek(X%));" "
      ;CHR$(FNgreek(X%+32));"  "A$;
170   PRINTTAB(22)CHR$(FNgreek(X%+12))
      ;" ";CHR$(FNgreek(X%+44));"  "B$
180   NEXT
190 VDU28,0,31,39,27
200 COLOUR129:COLOUR0:CLS
210 PRINT"Lets say that each letter of
     the Greek"'"Alphabet corresponds
    to a letter in the"'"English alpha
    bet:"''"   PRESS RETURN TO CONTINU
    E";
220 REPEATUNTILGET=13
225 *FX202,48
230 REPEAT
240   CLS
```

# The Greek Alphabet

| | | |
|---|---|---|
| Α | α | alpha |
| Β | β | beta |
| Γ | γ | gamma |
| Δ | δ | delta |
| Ε | ε | epsilon |
| Ζ | ζ | zeta |
| Η | η | eta |
| Θ | θ | theta |
| Ι | ι | iota |
| Κ | κ | kappa |
| Λ | λ | lambda |
| Μ | μ | mu |
| Ν | ν | nu |
| Ξ | ξ | xi |
| Ο | ο | omicron |
| Π | π | pi |
| Ρ | ρ | rho |
| Σ | σ | sigma |
| Τ | τ | tau |
| Υ | υ | upsilon |
| Φ | φ | phi |
| Χ | χ | chi |
| Ψ | ψ | psi |
| Ω | ω | omega |

Type in Greek/English ? (G/E)

```
250   PRINT''"Type in Greek/English ? (
      G/E)";
260   REPEATA$=GET$:UNTILINSTR("GgEe",
      A$)>0
270   PRINT''"Enter String"'
280   Z$="":Y$=""
290   IFA$="G"ORA$="g" THENPROCgreek E
      LSEINPUTY$:FORX%=1TOLENY$:Z$=Z$+
        CHR$(FNgreek(ASCMID$(Y$,X%,1))
        ):NEXT
300   CLS:PRINT''"Greek  : ";Z$''"Engl
      ish: ";Y$
310   REPEATUNTILGET=13
320   UNTILFALSE
330 END
340 :
350 DATAalpha,nu,beta,xi,gamma,omicron
    ,delta,pi,epsilon,rho,zeta,sigma,e
    ta,tau,theta,upsilon,iota,phi,kapp
    a,chi,lambda,psi,mu,omega
360 :
370 DEFPROCdbl(chr)
380 ?&D00=chr:X%=0:Y%=&D:A%=&0A
390 CALL&FFF1
400 VDU23,245,?&D01,?&D01,?&D02,?&D02,
    ?&D03,?&D03,?&D04,?&D04
410 VDU245,8,10
420 VDU23,245,?&D05,?&D05,?&D06,?&D06,
    ?&D07,?&D07,?&D08,?&D08
430 VDU245
440 ENDPROC
450 :
460 DEFPROCgreek
470 REPEAT
480   A$=GET$
490   X%=FNgreek(ASCA$)
500   PRINTCHR$X%;
510   IFX%=127ANDLENZ$>0 THENZ$=LEFT$(
      Z$,LENZ$-1):Y$=LEFT$(Y$,LENY$-1)
       ELSEIFX%<>127 THENZ$=Z$+CHR$X%:
      Y$=Y$+A$
520   UNTILX%=13:Z$=LEFT$(Z$,LENZ$-1):
    Y$=LEFT$(Y$,LENY$-1)
530 ENDPROC
540 :
550 DEFFNgreek(e%)
560 IFe%<97ORe%>120 THEN610
570 IFe%=111 THEN=e% ELSEIFe%=113 ORe%
    =120 THEN=e%-1
580 IFe%=110 THEN=237 ELSEIFe%=112 THE
    N=238 ELSEIFe%=109 THEN=118
590 IFe%<109 THEN=e%+127 ELSE=e%+125
600 :
610 IFe%=67 THEN=246 ELSEIFe%=68 THEN=
    247
620 IFe%=72 THEN=249 ELSEIFe%=75 THEN=
    248
630 IFe%=78 THEN=250 ELSEIFe%=82 THEN=
    252
640 IFe%=85 THEN=253 ELSEIFe%=80 THENe
    %=251
650 IFe%=89 THEN=254 ELSEIFe%=70 THEN=
    90
660 IFe%=87 THEN=255 ELSEIFe%=88 THEN=
    254
670 IFe%=71 ORe%=74 ORe%=76 ORe%=77 OR
    e%=83 THEN=e%+1
680 IFe%=81 THEN=80 ELSEIFe%=84 THEN=8
    9
690 IFe%=86 THEN=88
700 =e%
710 :
720 DEFPROCchars
730 VDU23,224,&3100;&CC4A;&4ACC;&33;
740 VDU23,225,&4478;&4478;&7C42;&4040;
750 VDU23,226,&4204;&14A2;&2818;&1028;
760 VDU23,227,&201C;&810;&4234;&1C22;
770 VDU23,228,&7800;&7880;&8080;&7C;
780 VDU23,229,&3846;&2010;&1C20;&C02;
790 VDU23,230,&4C00;&22B2;&2222;&202;
800 VDU23,231,&120C;&4E22;&4472;&3048;
810 VDU23,232,&3000;&1010;&1410;&18;
820 VDU23,233,&C600;&5048;&5870;&464D;
830 VDU23,234,&2810;&808;&341C;&265;
840 VDU23,235,&2200;&4422;&B2CC;&8080;
850 VDU23,237,&F04C;&3820;&BC40;&C62;
860 VDU23,238,&3E00;&1454;&1414;&6;
870 VDU23,239,&3E00;&8850;&9088;&60;
880 VDU23,240,&3E00;&848;&A00;&4;
890 VDU23,241,&6400;&6262;&6C62;&30;
900 VDU23,242,&1C10;&5232;&9894;&1070;
910 VDU23,243,&4700;&18AC;&3518;&E2;
920 VDU23,244,&808;&AA49;&1C2A;&808;
930 VDU23,246,&7E00;&6060;&6060;&60;
940 VDU23,247,&1800;&2C2C;&4646;&FF83;
950 VDU23,248,&1000;&2828;&4444;&C6;
960 VDU23,249,&3800;&8244;&82BA;&3844;
970 VDU23,250,&FE00;&82;&38;&FE82;
980 VDU23,251,&FE00;&4444;&4444;&44;
990 VDU23,252,&FE00;&6082;&6010;&FE82;
1000 VDU23,253,&1038;&927C;&7C92;&3810;
1010 VDU23,254,&4438;&8282;&2844;&EEAA;
1020 VDU23,255,&9238;&5454;&1038;&3810;
1030 ENDPROC
```

# BEGINNERS AND LISTINGS

One of the earliest ways to get a grasp of programming is by typing in programs from magazines. In such cases, it isn't too important how good the game is, or how well the program works, because you are just trying to get a feel for the language.

Games programs are often more exciting in their description than in their execution, but they have an advantage over other programs - you don't have to know what you are doing to work them, so it's easy to find out if you've typed it in correctly or not. This is because in general you only need a few keys to operate the program. (left, right, kill) Any bugs are quickly found, because you can use the program easily.

## How to type them in?

For a start, all those line numbers, increasing in steps of 10 (usually).
AUTO is an AUTOmatic line-numbering command.
Type in AUTO (and press RETURN), and each time a number appears, press Return again. The next number appears, 10 higher than the previous one. These are your line numbers.
You can vary this - try `AUTO5,5` or `AUTO 100,1`

## SPACES

What about all those spaces? They are hard to count sometimes. If the spaces come straight after the line number, ignore them. In fact, if you are using AUTO, a space automatically appears. So, ignore any spaces between line number and the command of the program.

If the spaces come inside the commands they may occasionally be necessary eg

```
10PRINT"These   4 spaces were meant"
```

but

```
20 UNTIL X% = 0     is the same as   20UNTILX%=0
```

This leads one to suspect that spaces are for legibility; to ease reading the program.

## LISTO

is another command to improve legibility in a program listing, but it is like AUTO, in that it isn't used within a program. It is something you type in when you want this facility.
LISTO allows the listing to be printed out (on screen and printer) in varying formats.
LISTO0 means that the listing will be printed with no spaces between the line number and the program command:

```
10PRINT"arfle"
```

LISTO1 means that there will be one space put between the line number and command:

```
10 PRINT"arfle"
```

Beeblet usually uses LISTO7. This not only puts a space between the line number and command, it also shows where the loops within the program occur, by indenting them.

Look at p.22 of May '85 'Beeblet'. The listing begins with a REPEAT, and every line after it is indented 2 spaces until the UNTIL FALSE cancels it.
 If you have March '85 Beeblet, look at p.5 for a blatant example of indenting loops. All the commands following each FOR command are indented a further 2 spaces, until each NEXT command signals the end of a loop, and the end of its indenting.

These spaces are a simple way of showing where a program has to loop round. The advantage of this facility is that when you are looking through a listing, what is within the loop is almost like text being put in brackets - it is a deeper level of detail than the surrounding program. You can skip over it paying it less attention if you wish. Such spaces will not appear in your listing on the screen unless you demand it by use of LISTO, or unless you type them in. If you haven't actually typed in the spaces, they will not be included when you save the program.

Note that the only "looping" command that does not show up through indenting is Goto. Read the User Guide cautions.

## FUNCTION KEY LIST

After LISTO you also need to type in LIST. Here is a handy function key definition to use:

```
*KEY0MODE6|MLISTO7|MLIST|N|M|O
```

Now every time you want a listing, just press functionkey 0, and up comes the listing in Mode 6. Of course, if you prefer some other mode, type in the one of your choice eg *KEY0MODE3...

The |N|M|O at the end of the definition make the listing pause at each screenful (paging mode), but go out of paging mode when you have looked through the entire listing. If you ESCAPE to get out of the listing, you will need to press Control and O together to cancel the paging mode.

## DEALS AND DISCOUNTS

Send stamped addressed envelope for details of the deals negotiated for UG members. Please mark your request for the attention of the Equipment Officer, C/O the UG Box 9592 Wgtn

# WINDOWS

Tim Ryan

This is a program to allow up to eight text and graphics windows to be defined simultaneously. It stores all the definitions in memory, and on the receipt of the command to swap, copies the new defintion to OSWRCH.

To use it is very simple.
1. Turn MULTI-WINDOW on by:

```
CALL &1200
```

2. Define windows to be used:

```
VDU 0,24,0-7,X;Y;X1;Y1;
VDU 0,28,8-15,X,Y,X1,Y1
```

3. Turn on required windows

```
VDU 0,3:VDU 0,11
```

4. When finished turn MULTI-WINDOW off

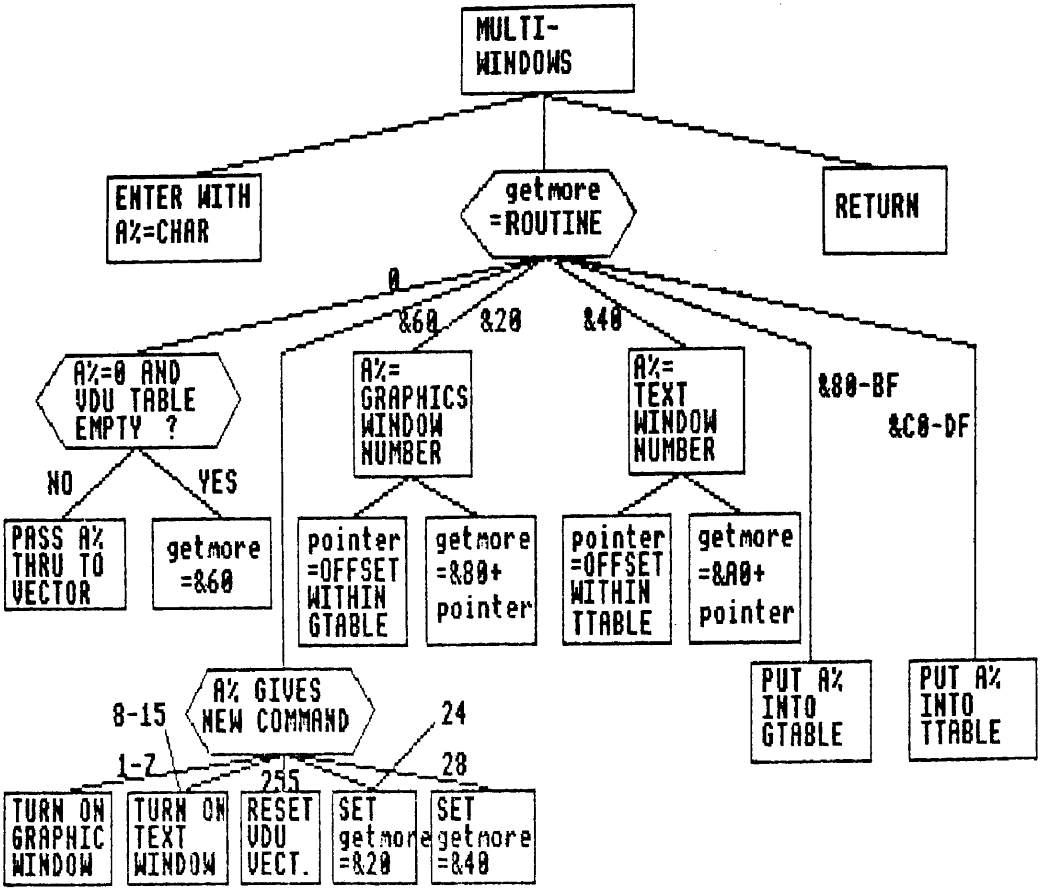
```
VDU 0,255
```

Note the graphics windows are numbered from 0 to 7 and the text windows are 8 to 15. This is used to allow a graphics and a text window to be in use simultaneously.

There is one minor niggle with the program as it is now: as you swap from window to window, the routine does not keep track of the TAB positions for each window separately.
There are two ways around this.
1. Add assembly code to store the current TAB positions just before changing windows.
2. Do the same in BASIC.

The program works by intercepting the OSWRCH vector, so it is quite important to switch the system off when no longer in use. I also suggest doing so in an ON ERROR statement
`ON ERROR VDU 0,255`
However, it CAN be left on without disturbing things, the main problem that does arise for some reason is when loading a new program.

The example program shows the system in use by printing numbers in four text screens. Note the number is printed on the bottom line of the window to force scrolling to occur.



```
 10 REM MULTI-WINDOW BY TIM RYAN
 20 WRCHV=&20E:IND3V=&234:OSBYTE=&FFF4
 30 Astore=&F8:getmore=&F9
 40 start=&1200:Gwindow=start+&110:Twi
    ndow=start+&150
 50 FOR assembly=0 TO 2 STEP 2
 60   P%=start
 70   [ OPT assembly
 80 \
 90 \ set up VDU vector to point to wi
             ndow program
100 \ IND3V gets old vector, for progr
             am to use
110 \
120 .start    LDA #&4C
              STA IND3V-1
130           LDA WRCHV
              STA IND3V
              LDA WRCHV+1
              STA IND3V+1
140           LDA #window MOD 256
              STA WRCHV
              LDA #window DIV 256
              STA WRCHV+1
150           LDA #0
              STA Astore
              STA getmore
160           RTS
170 \
180 \ save the VDU char in an unused p
              art of system memory in
              page 0
190 \
200 .window   STA Astore
210 \
220 \ getmore=0 and VDU code=0 means n
              ew multi-window command
```

```
230 \ getmore<>0 means multi-window co
              mmand in progress
240 \ getmore=0 and VDU code<>0 means
              standard VDU command
250 \
260           LDA getmore
              BNE command
270           LDA Astore
              BNE passthru
280           LDA #&DA
              LDX #0
              LDY #&FF
              JSR OSBYTE \ get VDU que
              ue length to see if VDU
              0 is ours
290           CPX #0
              BNE passthru
300           LDA #&60
              STA getmore
              RTS
310 \
320 .passthru LDA Astore
              JMP (IND3V)
330 \
340 \ the next number in the VDU strin
              g is looked at now
350 \ its meaning depends on the value
               in getmore
360 \ getmore=&60 means new command
370 \ getmore=&20 means a graphics win
              dow is being defined
380 \ getmore=&40 means a text window
              is being defined
390 \ getmore=&8x means the coords for
               a graphics window are b
              eing entered
400 \ getmore=&Cx means the coords for
               a text window are being
               entered
410 \ note the x after &8 and &C. This
               is because the low bits
               of getmore are
420 \ used as a pointer to the positio
              n in mem to store the wi
              ndow definition
430 \
440 .command  CMP #&20
              BEQ getGW
450           CMP #&40
              BEQ getTW
460           CMP #&60
```

```
              BEQ newcom
470           AND #&C0
480           CMP #&80
              BEQ GW
490           CMP #&C0
              BEQ TW
500 .error    BRK
              BRK
              BRK
510 \
520 \ user does not want multi-window
              anymore. reset VDU vecto
              r
530 \
540 .turnoff  LDA IND3V
              STA WRCHV
              LDA IND3V+1
              STA WRCHV+1
              RTS
550 \
560 \ all we've received so far is the
               VDU 0 to signal start o
               f a new command
570 \ VDU char=24 means define graphic
              s window
580 \ VDU char=28 means define text wi
              ndow
590 \ VDU char=FF means quit - reset V
              DU vector
600 \ VDU char=0-7 means turn on graph
              ics window
610 \ VDU char=8-15 means turn on text
               window
620 \
630 .newcom   LDA Astore
640           CMP #24
              BEQ newGW
650           CMP #28
              BEQ newTW
660           CMP #&FF
              BEQ turnoff
670           CMP #16
              BCS error
680           LDA Astore
690           CMP #8
              BMI setGW
              BPL setTW
700 \
710 \ the first byte of instruction wa
              s a define window char
```

```
720 \ set a flag to get window number
              to redefine from next VD
              U char
730 \
740 .newGW    LDA #&20
              STA getmore
              RTS
750 .newTW    LDA #&40
              STA getmore
              RTS
760 \
770 \ this byte says which window is t
              o be defined
780 \ use it to index into the table
790 \
800 .getGW    LDA Astore
810           ASL A
              ASL A
              ASL A
820           ORA #&80
              STA getmore
830           RTS
840 \
850 .getTW    LDA Astore
              SEC
              SBC #8
860           ASL A
              ASL A
870           ORA #&C0
              STA getmore
880           RTS
890 \
900 \ copy VDU chars to window definit
              ion table
910 \ uses low bits of getmore as poin
              ter within table
920 \
930 .GW       LDA getmore
              AND #&3F
              TAY
940           LDA Astore
              STA Gwindow,Y
950           TYA
              AND #7
              CMP #7
              BPL done
960           INC getmore
970           RTS
980 \
990 .TW       LDA getmore
              AND #&3F
              TAY
```

```
1000          LDA Astore
              STA Twindow,Y
1010          TYA
              AND #3
              CMP #3
              BPL done
1020          INC getmore
1030          RTS
1040 \
1050 \ the command was to set up a wind
              ow number
1060 \ copy the appropriate number of b
              ytes to the original VDU
               vector
1070 \ so that the new window will be a
              ctioned
1080 \
1090 .setGW   LDA #24
              JSR IND3V-1
1100          LDA Astore
1110          ASL A
              ASL A
              ASL A
1120          TAY
1130          LDX #8
1140 .loadG   LDA Gwindow,Y
              JSR IND3V-1
1150          INY
              DEX
              BNE loadG
1160          BEQ done
1170 \
1180 .setTW   LDA #28
              JSR IND3V-1
1190          LDA Astore
              SEC
              SBC #8
1200          ASL A
              ASL A
1210          TAY
1220          LDX #4
1230 .loadT   LDA Twindow,Y
              JSR IND3V-1
1240          INY
              DEX
              BNE loadT
1250 \
1260 .done    LDA #0
              STA getmore
              RTS
1270          ]
1280   NEXT
```

```
************************************

        L.WNDWTST

************************************

 10 REM MULTI-WINDOW DEMONSTRATION BY
    TIM RYAN
 20 REM THERE ARE 8 GRAPHICS AND 8 TEX
    T WINDOWS, NUMBERED 0-7 AND 8-15
 30 REM WE ARE USING TEXT WINDOWS 8-13
     AND GRAPHICS WINDOWS 6 AND 7 ONLY
 40 REM THE WINDOWS ARE DEFINED NEARLY
     NORMALLY:
 50 REM VDU 0,24/28,window-num,X1,Y1,X
    2,Y2 - NOTE USE SEMICOLONS FOR GRA
    PHICS
 60
 70 MODE 1
 80 *LOAD WNDWCDE
 90 ON ERROR VDU 0,255,26,20:END
100 COLOUR 129:CLS
110
120 REM FIRE UP MULTI-WINDOW PROGRAM
130
140 CALL &1200
150
160 VDU 23,1;0;0;0;0
170
180 REM DEFINE ALL THE WINDOWS THAT AR
    E NEEDED FOR THE DEMO
190
200 VDU 0,28,8,5,16,14,10
210 VDU 0,28,9,25,16,34,10
220 VDU 0,28,10,5,27,14,21
230 VDU 0,28,11,25,27,34,21
240 VDU 0,28,12,5,4,34,1
250 VDU 0,28,13,7,30,34,30
260 VDU 0,24,6,530;480;740;780;
270 VDU 0,24,7,530;130;740;440;
280
290 REM SET COLOURS, FIRE UP TEXT WIND
    OW 12, AND PRINT MESSAGE
300
310 COLOUR 0:COLOUR 131:VDU 0,12:CLS
320 PRINT '"  MULTI-WINDOW DEMONSTRATI
    ON"
330
340 REM SET UP FOUR TEXT WINDOW BACKGR
    OUNDS
350
360 COLOUR 130:COLOUR 0
370 FOR T=8 TO 11
380   VDU 0,T:CLS
390   NEXT
400
410 REM SET UP BORDERS FOR THE TWO GRA
    PHICS WINDOWS
420
430 GCOL 0,0:MOVE 535,476:DRAW 535,784
440 DRAW 745,784:DRAW 745,476:DRAW 535
    ,476
450 MOVE 535,126:DRAW 535,444
460 DRAW 745,444:DRAW 745,126:DRAW 535
    ,126
470 GCOL 0,3
480
490 REM CYCLE THROUGH THE EIGHT WINDOW
    S DEFINED, PRINTING OR DRAWING
500
510 FOR count=0 TO 100
520   COLOUR 0:COLOUR 130
530   FOR T=6 TO 13
540     VDU 0,T
550     X1=RND(400)+440:X2=RND(400)+44
        0
560     Y1=RND(800)+100:Y2=RND(800)+10
        0
570
580     REM NOTE EACH WINDOW REMAINS D
        EFINED TILL THE NEXT IS CALLED
590
600     IF T<8 THEN MOVE X1,Y1:DRAW X2
        ,Y2
610     IF T<12 AND T>7 THEN PRINT TAB
        (4,6);count+100*(T-8)
620     IF T=12 THEN COLOUR 1:COLOUR 1
        31:PRINT TAB(8,2) "T*count = "
        ;T*count;
630     IF T=13 THEN COLOUR 2:COLOUR 1
        29:PRINT "X1=";X1;" Y1=";Y1;"
        X2=";X2;" Y2=";Y2;
640     NEXT
650   NEXT
660
670 REM DEMO ENDED TURN OFF MULTI-WIND
    OW AND RESET TO STANDARD SIZES
680
690 VDU 0,255,26,20
```

# The BBC Prestel Adapter

Simply and easily connected to a BBC Model B Microcomputer, the Prestel Adaper converts the computer into a powerful terminal capable of two way communication via telephone lines. Provides access to armchair shopping, Business information etc.

**Features.**

1. The Presetl Adapter enables users to communicate, through the Post Office Telephone Network, with various information services and databases in NZ and Overseas.

2. The Aditel System for example provides subscribers with a wide range of constantly updated information.

3. Prestel allows the BBC Model B to "talk" to almost any computer with a 1200/75 Bits per second modem.

4. Available databases include: Share and stock prices, weather, news , wool and livestock prices, farming information etc etc. More databases are constantly being added.

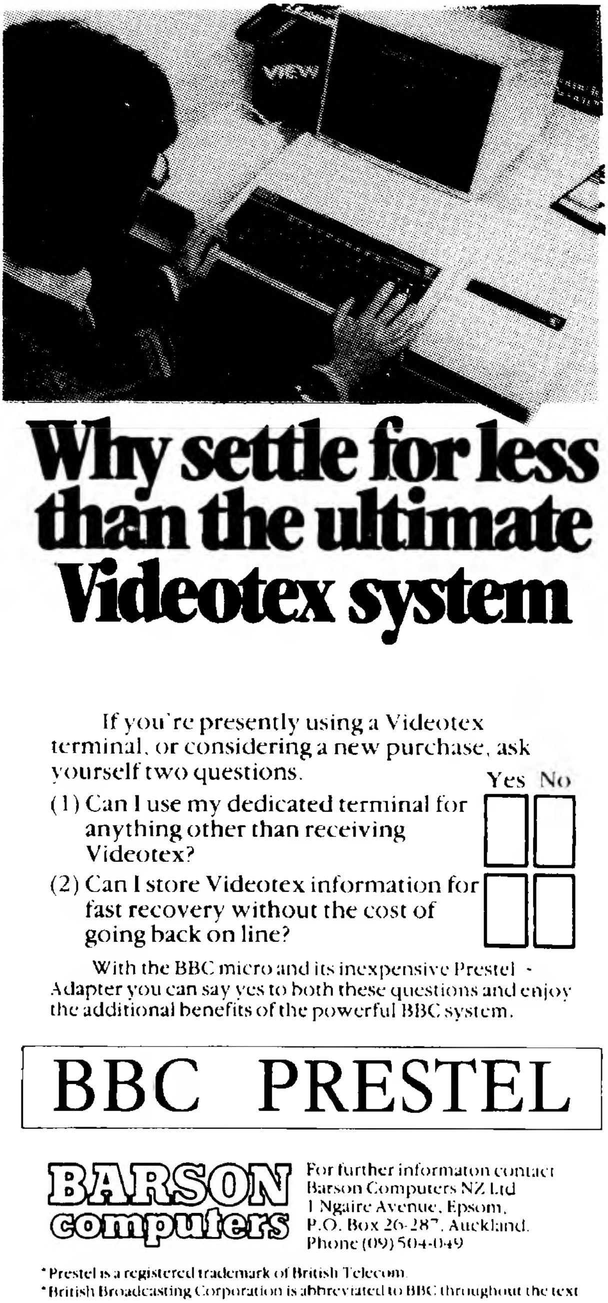
5. The Prestel Adapter allows the user to edit text until satisfactory and then send the completed messages by telephone. The system is fast, inexpensive and secure.

6. Information received from Databases can be saved to disc or cassette or dumped to a printer.

**Who Needs The Prestel Adapter?**

Anyone with a need for access to a wide variety of information, particularly farmers, travel agents etc. Prestel is a simple, inexpensive way of joining an rapidly expanding network of databases.

**ASK ABOUT THE NEW BBC FARMPACK**



# Why settle for less than the ultimate Videotex system

If you're presently using a Videotex terminal, or considering a new purchase, ask yourself two questions.

| | Yes | No |
|---|---|---|
| (1) Can I use my dedicated terminal for anything other than receiving Videotex? | ☐ | ☐ |
| (2) Can I store Videotex information for fast recovery without the cost of going back on line? | ☐ | ☐ |

With the BBC micro and its inexpensive Prestel Adapter you can say yes to both these questions and enjoy the additional benefits of the powerful BBC system.

## BBC PRESTEL

**BARSON computers**

For further informaton contact
Barson Computers NZ Ltd
1 Ngaire Avenue, Epsom,
P.O. Box 26-287, Auckland.
Phone (09) 504-049

* Prestel is a registered trademark of British Telecom
* British Broadcasting Corporation is abbreviated to BBC throughout the text

THE BBC MICROCOMPUTER
PRESTEL ADAPTER

AVAILABLE AT WHITCOULLS
AND ALL MAJOR BBC
RETAILERS

RETAIL PRICE $599-00

ADITEL SUBSCRIPTIONS
$180 PER ANNUM
+ 17c per minute

The PRESTEL adapter is a fully type approved 1200/75 baud direct autodialling modem and is supplied with PRESTEL software on ROM

For more information contact
BARSON COMPUTERS
1 NGAIRE AVE
EPSOM
AUCKLAND
Ph. (9) 504-049

# MACROS IN VIEW

by Duncan McEwan

The editor of Beeblet asked me if I could answer a question sent in by a member, concerning the use of macros in the View word processor. After experimenting for a while, I realised that although Views macros could be very useful, there are a couple of points that you have to watch out for. So as well as answering the question, I thought that an article about macros may be of use to some people.

Firstly what is a macro, and how can one be used in word processing. A macro is similar in many ways to a PROC in BBC Basic. Suppose you have a piece of text that is going to be repeated a number of times in a document. Rather that have the actual text repeated at each place in the document, you can define it in a macro, and at each point that you want the text to appear you "call" the macro. This has a number of advantages. If the repeated text is long, it will save space. Also, if you want to change the repeated text, you only have to change the macro definition.

In View, the macro facility is provided by "Stored Commands" (two character commands placed in the left margin, that cause View to perform certain functions as the text is being printed).

The DM (Define Macro) command indicates the begining of a macro, and gives the macro a name. The EM (End Macro) command marks the end of the definition. The name associated with the macro is now just like any other View stored command, and can be invoked in the same way. The following silly example illustrates these commands. The macro XX is defined, and then used twice interspersed with other text.

```
     DM XX
        This is a piece of text that will be repeated several
times in my document.
     EM
     {some other text might go here}
     XX

     {may be something else here as well}
     XX
```

From what has been said above, macros do not appear very usefu
 How often do you need to repeat exactly the same text severa
times in a document? They are more useful if, as in View, th

text of the macro can be made to vary slightly with each "call". This is done by allowing macros (like Basic PROC's) to be called with parameters. The macro can be invoked each time with different parameters.

In View, the text of the macro can contain the special "markers" @0 through to @9. The macro invocation can contain a comma separated list of up to 10 parameters. The marker @0 is replaced by the first parameter, @1 by the second, and so on. The (again, silly) example below shows how this works.

```
DM GB
   There are @0 green bottles hanging on the wall,
   There are @0 green bottles hanging on the wall,
   And if one green bottle should accidently fall,
   There'd be @1 green bottles, hanging on the wall.

EM
```

The macro can be invoked as many times as you want as follows.

```
GB one hundred, ninety nine
GB ninety nine, ninety eight
```

and so on.

A more useful use of macros with parameters would be in producing a form letter. The text of letter is always the same, except for the name and address of the person it is being sent to. The letter could be defined as a macro, with the name and address replaced by @0 and @1. The macro would be called as many times as required, with the different names and addresses passed as parameters.

Now, having briefly explained what macros are, and how they can be used, I will now look at some of the problems with View's macros. This first problem is the answer to the question I was originally asked (at least I think it is - assuming I interpreted the question correctly!).

Although View can be told to fill the text of a macro (ie fit as many full words on to a line as possible), when the macro is invoked, its text cannot become part of a larger section of filled text. There will always be a break before and after the text of the macro. That is, the following definition and text

```
DM XX
   This is the second sentence of the paragraph.
```

```
    EM
       This is the first sentence of the paragraph.
    XX
       And this is the third sentence.
```

cannot be made to produce

This is the first sentence of the paragraph. This is the second sentence of the paragraph. And this is the third sentence.

This is because all the text filling etc is done as the text is entered, whereas macros are processed as the text is being printed. So no text filling takes place when the text of the macro is inserted into the main text. The stored command used to invoke the macro must start a new line, and therefore, the macro text itself must. Similarly for the text immediately following the macro invocation.

A second problem that I encountered is also associated with text filling. When the macro text is typed in, View fills it according to the set line length. When the macro is called, if parameters are substituted for the words @0, @1, etc, the length of the line will probably become longer. Unfortunately View does not reformat it, so the length of the line produced will be longer than the other lines in the document, and in fact may be longer than the paper width!

That is all I have to say about View's macros. I hope that I answered the question that I was originally asked. If not, then maybe I at least gave some people an idea of what macros are, and how they can be used.

## BRIGHT IDEAS

Steve W.

Everyone knows how BASIC (even BASIC 2) is so inefficient at handling string storage. Why doesn't someone come up with a utility ROM which has a *GARBAGE command, which runs around reclaiming unused space, and resetting variable pointers etc?? Another useful command would be a *SORT command, which would be either memory based or disk based depending on parameters.

Beryl and Ron D. have indicated their displeasure at the "ugly" thick black lines at top and bottom of the page. Any other views?

# VARIABLES AND PROCEDURES

JL

The possibility of being able to use words instead of a letter for variables on the BEEB provides a chance for easily recognised and meaningful names throughout a program. This coupled with the use of PROCedures should make the deciphering of a program like to reading a good book and not an exercise in what you can remember while jumping from place to place. RIGHT? Right,well almost.

P.106 etc. of the U.G.describes the action of variables in through and out of PROCs but I got lost somewhere trying to follow it. Why did the writer think that the beginner would be able to grasp the fine points of variables,PROCs while leaping around in FOR loops. I misread it somewhere and as a result of this and a few typing errors spent hours trying to get a program working. In desperation I wrote a program similar to what follows to find out (if possible) how it all works.

It was hoped that once a variable label was decided on it would be possible to to keep it right throughout the program especially where it would be operated on by one or more PROCs. Variables established outside PROCs are known as GLOBAL while those defined inside PROCs are LOCAL and known only to the PROC. They can also be called LOCAL which means that a variable of the same name cannot be passed into the PROC. (For variable also read contents of variable where appropriate).

```
************************************

          L.PASS1

************************************

 10 REM  PASSING VARIABLES IN AND OUT
 20 REM          OF PROCEDURES
 25 REM            BY J.LOGIE
 30 ANN=11.5:G%=8:anton=5.8:neville%=5
    :A$="EDITOR"
 50 PROCContribute(ANN,G%,anton,nevill
    e%,A$)
 60 PRINT ANN,G%,anton,neville%,A$
 90 END
100 DEF PROCContribute(ANN,G%,anton,ne
    ville%,A$)
110 ANN=ANN+1:G%=G%+1:anton=anton+1:ne
    ville%=neville%+1:A$="A$+IAL"
200 ENDPROC
```

```
************************************

          L.PASS2

************************************

 10 REM  PASSING VARIABLES IN AND OUT
 20 REM          OF PROCEDURES
 30 ANN=11.5:G%=8:anton=5.8:neville%=5
    :A$="EDITOR"
 50 PROCContribute(ANN,G%,anton,nevill
    e%,A$)
 60 PRINT ANN,G%,anton,neville%,A$
 70 PRINT A,A%,a,a%,B$
 90 END
100 DEF PROCContribute(A,A%,a,a%,B$)
110 A=A+1:A%=A%+1:a=a+1:a%=a%+1:B$=B$+
    "IAL"
200 ENDPROC
```

# DECAY SERIES

Don has pointed out that the program published last month along with his voltage/ current measuring article was in fact the wrong program - yes Don, this had just dawned on me, in spite of it being called "Voltage", and being the only approriate sounding one I can locate. Next month I will publish the correct program.

Meanwhile, some explanation of the program that was published!

A radioactive atom decays by losing part of its nucleus. The new atom formed may also be radioactive and there may be a whole series of different atoms formed. The series finally ends when a stable isotope of lead is formed.

The more unstable atoms will have a shorter half-life which can vary between a fraction of a second and thousands of millions of years.

The program shows, with the aid of a bar graph, how the number of each type of atom changes with time. There are three types of radioactive atom whose half-lives can be chosen. The fourth atom is stable.

```
************************************
```

## L.PASS3

```
************************************

 10 REM  PASSING VARIABLES IN AND OUT
 20 REM           OF PROCEDURES
 30 ANN=11.5:G%=8:anton=5.8:neville%=5
    :A$="EDITOR"
 50 PROCContribute(ANN,G%,anton,nevill
    e%,A$)
 60 PRINT ANN,G%,anton,neville%,A$
 70 PRINT A,A%,a,a%,B$
 90 END
100 DEF PROCContribute(A,A%,a,a%,B$)
110 ANN=A+1:G%=A%+1:anton=a+1:neville%
    =a%+1:A$=B$+"IAL"
200 ENDPROC
```

```
************************************
```

## L.PASS4

```
************************************

 10 REM  PASSING VARIABLES IN AND OUT
 20 REM           OF PROCEDURES
 30 ANN=11.5:G%=8:anton=5.8:neville%=5
    :A$="EDITOR"
 50 PROCContribute(ANN,G%,anton,nevill
    e%,A$)
 60 PRINT ANN,G%,anton,neville%,A$
 70 PRINT A,A%,a,a%,B$
 90 END
100 DEF PROCContribute(ANN,G%,anton,ne
    ville%,A$)
110 ANN=ANN+1:G%=G%+1:anton=anton+1:ne
    ville%=neville%+1:A$=A$+"IAL"
120 A=ANN:A%=G%:a=anton:a%=neville%:B$
    =A$
200 ENDPROC
```

# REVIEW OF OXFORD PASCAL (BBC)

Review copy supplied by Whitcoulls.By Odie.

PASCAL
This comes in a neatly packaged box which contains a ROM, a Manual and a disc with two Compilers , a Linker , a Locater and a few example programs on it. This is a complete implementation of the language PASCAL with a few extra features added to make full use of the BEEB's graphics and sound commands and the built in assembler. This product is not TUBE compatible. The Manual itself is very clear and easy to follow with many examples, including a listing of the program NIM ( Included on the disc ). The index is not as good as it could have been, as it only includes the key words. The error messages that can occur are clear and meaningful.

The first compiler on the disc is one which is loaded in initially ( the Resident compiler ) and the programs are compiled entirely within memory, thus limiting the length of the program. (About 15K is available to the user with this option ). The second compiler on the disc is one which resides on the disc permanently. It is called after the program has been saved using the command COMP <filename>. This option allows all the free memory to be used depending on the mode you are in. The Linker allows you to join or merge small modules of code together to form one large program. The Locater is a utility which allows the code generated by the compiler to be run on any BBC with or without the Pascal ROM fitted, without the user even understanding Pascal at all.

The Editor Commands:

The Editor works in much the same way as when you are writing a basic program ( i.e. a line editor ). You enter each line separately with only one statement per line.
The Editor commands are :-
BASIC( To return to BASIC )
AUTO( Same as in BASIC )
LIST( Same as in BASIC )
NEW, OLD( Same as in BASIC )
DISK( To enter the Disk compiler )
RESIDENT( To reload the Resident compiler )
RENUMBER( Same as in Basic )
FIND( Search for a specified string )
CHANGE( Search and replace a string )
DELETE( Same as in BASIC )
These next three only work in the RESIDENT compiler moce.

R ( RUN )( Run the program. Compiling if necessary )
L( Compile and display the program on VDU )
P( Compile and list on Printer )
These two only work in the DISC compiler mode.
COMP <filename>,option( Compiles a program on disc )
EX <filename>( To Execute an object file )

The second parameter in the COMP command can be used to list the program on the VDU or the Printer or to select the drive to save to.

The Pascal Language itself is very similar to BBC Basic but has many extra feature such as being able to define your own variable types i.e. not just integer, real and string ( See the example program). It includes structures such as REPEAT UNTIL, WHILE - DO, CASE, IF THEN ELSE, FOR NEXT, PROCEDURES and FUNCTIONS. (This is where BBC Basic got its procedures and functions from.)

The Special Graphics and Sound Commands:

| BASIC | PASCAL |
|---|---|
| MODE k | MODE(k) |
| DRAW x,y | DRAW(x,y) |
| MOVE x,y | MOVE(x,y) |
| PLOT k,x,y | PLOT(k,x,y) |
| GCOL d,e | GCOL(d,e) |
| VDU 31,x,y,asc('a') | VDU(x,y,'a') |
| SOUND a,d,c,d | SOUND(a,b,c,d) |
| OSCLI expression | WRITE(OSCLI,(expression)) |
| GET | GETKEY |
| ENVELOPE a,b,c,d,e,f g,h,i,j,k,l,m,n | ENVELOPE(a,b,c,d,e,f,g,h, i,j,k,l,m,n) |

There are also a few miscellaneous commands.
PEEK a location in memory.
POKE a location in memory.
OSVDU This is used to pass the rest of the VDU commands to the BEEB instead of just VDU31.

The language also has its own filing system commands PUT and GET for SAVE and LOAD and includes advanced file handling in the form of POINTERS used to read specific variables from a file.

EXAMPLE PROGRAM

This program is unchanged from the original which was written on the DEC VAX 11/750 computer at Victoria University (All programs which I wrote during my Varsity career also ran unchanged, including some which were extremely long and complex).

```
 10 PROGRAM CONVERT;
 15 (* Comments all enclosed in Brackets with an asterisk as can
be seen *)
 20 (* This program converts all text entered to upper *)
 30 (* Case leaving numbers alone. *)
 40  VAR ch : CHAR;    (* CHAR is the variable type for
characters. *)
 50                    (* ch is the character read from the
keyboard. *)
 60 BEGIN
 70   WRITELN ('Please type in your message.');
 80           (* This is the same as PRINT in Basic. *)
 90   WRITELN;
100
110   WHILE NOT eof DO    (*Dountilescapeis pressed. *)
120    BEGIN           (* The begin of the WHILE loop. *)
130
140     WHILE NOT eoln DO (* Do until RETURN is pressed. *)
150      BEGIN
160         read (ch);      (* Same as INPUT or GET in Basic.
*)
170       IF ( ch >= 'a') AND ( ch <= 'z')
180        THEN    (* Only if ch is in lower case. *)
190         ch := chr ( ord(ch) - ord('a') + ord('A'));
200                   (* ORD is the same as ASC('ch') in Basic.
*)
210              (* CHR is the same as CHR$ in Basic. *)
220          WRITE ( ch )    (* WRITE is the same as PRINT;. *)

230       END;     (* Of the eoln loop. *)
240
250      WRITELN;  (* To start printing on the next line. *)
260        READLN       (* To skip over the RETURN code. i.e.
CHR$13. *)
270
280    END   (* Of the eof loop. *)
290 END.     (* Of the program. *)
```

In summary I found this to be a very professional product with all the features that are now standard in PASCAL. The fact that it does not work with the 6502 Second Processor ( I have one of these ) means I will not be buying it. At $179 I recommend it to Pascal and Basic users alike as it is far superior to Basic and very easy to learn.

Next month a comparitive review of Acornsoft ISO PASCAL and OXFORD PASCAL both of which are not written by GCF......!!!!!!.

# String Handling Package

The package consists of a set of assembler routines defined in an initialisation procedure, and a set of BASIC interface functions. These are combined into a chunk of BASIC, which is included in your program. The program calls the initialisation procedure once, and can then use the interface functions as required.

The package is supplied in two forms, as a BASIC program, "STRING", and as an ASCII file, "STRINGA", created by *SPOOL for use with *EXEC (See the User Guide, p402). ( Editor's note : There is no point in me printing both "versions".) It is numbered from 32000 upward, which should be well above other line numbers used. To include it in an existing program use "STRINGA" as described in the Guide, to use it in a new program, LOAD "STRING" before you start typing lines.

Somewhere near the start of your program, where it will be executed before any use is made of the package, you must include the statement PROCstrinit. This will set the package up, and you may now use the functions provided as appropriate. Please note that PROCstrinit uses the global variable strent to keep track of the machine code, you must not use it for anything else.

The routines make use of zero-page locations &70 through &7B for working storage. Anything kept in these locations may be corrupted during a call. The routines do not require that these locations be kept free between calls, they may be freely used by other routines you may include, as long as these routines do not expect exclusive use of the storage.

Functions available:

FNbefore(P$,C$)
Returns P$ up to but not including the first occurrence in P$ of a character also in C$.
Example:FNbefore("THREE WORD LINE"," ") gives a result of "THREE"

FNafter(P$,C$)
Returns P$ starting at the first character in P$ that is not in C$.
Example:FNafter(".bc.,"," .,") gives a result of "bc.,"

FNwhile(P$,C$)
Rreturns P$ up to but not including the first character in P$ that is not in C$.
Example:FNwhile("4096KB","0123456789") gives a result of "4096"

FNfrom(P$,C$)
Returns P$ starting at the first character of P$ that is also in C$.
Example:FNfrom("XYZZY","NZ") gives a result of "ZZY"

FNreverse(P$)
Returns P$ reversed in order.
Example:FNreverse("abc") gives a result of "cba"

FNtranslate(P$,F$,T$)
Translates P$ according to F$ and T$. Each character of P$ that occurs in F$ is replaced by the corresponding character of T$. If F$ and T$ are of equal length, then characters not in F$ are unaltered, but if T$ is longer than F$ then characters not in F$ are replaced by the first character of T$ that does not correspond to a character in F$. T$ must not be shorter than F$, or an error will occur. F$ should not contain duplicate characters.
Example:FNtranslate("ABCDE","BC","XY")
gives a result of "AXYDE"

FNupcase(P$)
Translates any lower case alphabetic characters in P$ to upper case. No other characters are changed.
Example:FNupcase("The Larch!") gives a result of "THE LARCH!"

The only error message produced is "String Error", which is normally caused by calling FNtranslate with the third parameter shorter than the second one. The message may also be produced if you try to bypass the BASIC functions and CALL the machine code yourself, and get the call sequence wrong. I urge you not to try this, as the BASIC does do other important things besides CALL the machine code.

```
32000 DEFFNbefore(P$,C$):LOCAL A%:A%=0
32010 IF P$="" OR C$="" THEN =P$
32020 CALL strent,P$,C$
32030 =LEFT$(P$,A%)
32040 DEFFNafter(P$,C$):LOCAL A%:A%=0
32050 IF P$="" OR C$="" THEN =P$
32060 CALL strent+3,P$,C$
32070 =MID$(P$,A%+1)
32080 DEFFNwhile(P$,C$):LOCAL A%:A%=0
32090 IF P$="" OR C$="" THEN =""
32100 CALL strent+3,P$,C$
32110 =LEFT$(P$,A%)
32120 DEFFNfrom(P$,C$):LOCAL A%:A%=0
32130 IF P$="" OR C$="" THEN =""
32140 CALL strent,P$,C$
32150 =MID$(P$,A%+1)
32160 DEFFNupcase(P$)
32165 IF P$="" THEN =P$
32170 CALL strent+6,P$
32180 =P$
32190 DEFFNreverse(P$)
32195 IF LEN(P$)<2 THEN =P$
32200 CALL strent+9,P$
32210 =P$
32220 DEFFNtranslate(P$,F$,T$)
32225 IF P$<>"" AND T$<>"" AND F$="" THE
      N =STRING$(LEN(P$),LEFT$(T$,1))
32230 IF P$="" OR F$="" THEN =P$
32240 CALL strent+12,P$,F$,T$
32250 =P$
32260 DEFPROCstrinit
32270 LOCAL E1,E2,EU,ER,ET,L11,L12,L21,L
      22,LV1,LVS,LS1,LSX,LS2,LSL,LT1,LT2
      ,LT3,LU1,LR1,LVF,S1,S2,S3,I%,P%
```

```
32280 S1=&70:S2=&71:S3=&73
32290 DIM strent 310
32300 FOR I%=0 TO 2 STEP 2
32310   P%=strent:[OPT I%
32320             JMP E1
                  JMP E2
                  JMP EU
                  JMP ER
                  JMP ET
32330 .E1 JSR LV1
                  .L11 JSR LS1
                  BEQ L12
                  JSR LS2
                  BNE L11
                  LDY S1
                  DEY
                  .L12 STY &404
                  RTS
32340 .E2 JSR LV1
                  .L21 JSR LS1
                  BEQ L22
                  JSR LS2
                  BEQ L21
                  LDY S1
                  DEY
                  .L22 STY &404
                  RTS
32350 .LV1 LDA &600
                  CMP #2
                  BNE LVF
                  LDX #0
                  STX S1
                  INX
                  JSR LVS
                  LDX #4
                  JSR LVS
                  RTS
32360 .LVS LDA &600,X
                  STA S2
                  LDA &601,X
                  STA S2+1
                  LDA &602,X
                  CMP #129
                  BNE LVF
                  LDY #0
                  LDA (S2),Y
                  STA S3-1,X
                  INY
                  LDA (S2),Y
                  STA S3,X
                  LDY #3
                  LDA (S2),Y
                  STA S3+1,X
                  RTS
32370 .LS1 LDY S1
                  CPY S3+2
                  BEQ LSX
                  LDA (S3),Y
                  INY
                  STY S1
                  .LSX RTS
32380 .LS2 LDY #0
                  .LSL CMP (S3+3),Y
                  BEQ LSX
                  INY
                  CPY S3+5
                  BNE LSL
                  CPY #0
                  RTS
32385 .LVF BRK
                  ]:?P%=90:P%=P%+1:$P%="St
              ring Error"+CHR$0:P%=P%+13:[OPT
                  I%
32390 .ET LDA &600
                  CMP #3
                  BNE LVF
                  LDA #2
                  STA &600
                  JSR LV1
                  LDX #7
                  JSR LVS
                  LDA #0
                  STA S2+1
                  LDA S3+8
                  CMP S3+5
                  BEQ LT1
                  BCC LVF
                  LDY S3+5
                  LDA (S3+6),Y
                  STA S2
                  LDA #&FF
                  STA S2+1
32400 .LT1 JSR LS1
                  BEQ LSX
                  JSR LS2
                  BEQ LT2
                  BIT S2+1
                  BPL LT1
                  LDA S2
                  JMP LT3
                  .LT2 LDA (S3+6),Y
                  .LT3 LDY S1
                  DEY
                  STA (S3),Y
                  JMP LT1
```

Continued on Page 35

# TOP OF THE POPS!!

## WHAT YOU LIKED MOST FROM LAST YEAR'S 'BEEBLETS'

A collation of all the renewing surveys certainly made interesting and informative reading.

Question 1 "Name 3 articles or programs you appreciated"

- 14 said Modems
- 11 View Printer Driver
- 10 Disc drive series
- 10 1 MHz bus series (2 complaints - 1 not enough, 1 beyond comprehension)
- 10 Structured programming (3 disliked it for being too lengthy, simple)
- 7 Printer translations
  - relocating View
  - files series (4 complained too shallow / not complete / bit heavy / how set that pointer)
  - Circles & fast circles
- 5 Map of NZ
  - Solidisc review
  - disc space map
- 4 Easter
  - Elite review (1 complaint: too short, no screen shots)
  - tape-disc data transfer
  - Measuring Temperatures
  - screen dump in Basic

THANK YOU

therefore to Neil G, Robert N, Bryan H, Tony K (Akld), Anne F, Kerry K (Akld), Steve R, Anton E, Steve W (Akld), Keith F, GCF, Don A (Tokoroa), for these most popular articles.

THANKS ALSO for all these articles which got 2 or 3 mentions: Creed printer driver, Printer pounding, VSAS, Bitwise operators, Copyright, Italix, Aviator review, Disc menu, View to Basic, Local arrays, Lost memory, Hex on that Character, Random poetry, Pain in the neck, Advanced U.G. review, Co-processor, Hall of Fame, Teletext title page, Optical illusions, Osword from Basic, Hyper Hopper, Membership map, GAPII, 1985 Calendar, Multi-choice shell, User defined characters, DX, Left-Right frogs, Text display speed.

There were also many articles that got a single mention also - not quite every item published, but most of them got a mention, so thank you everyone that contributed.

Question 2: "What disappointed you?"
Apart from the contrasts noted above
several offered: those above my ability.
Anne's Complaint and Tim's Reasons both caught some flak as well as positive comment; so did arguments about sexism / sexist cartoons.
Perhaps the most important one perhaps was:
Barson's News - because it went away

Questions 5 & 6 asked what you looked for, and what you would like to see.
Several suggestions volunteered desired more local news, more on what is being done by other local groups, and just about everything under the sun.
The mix you were most interested in is tabled here:

| | |
|---|---|
| 123 | utility progs |
| 110 | 'useful' progs |
| 105 | reviews |
| 86 | learn m/code |
| 67 | hardware projects |
| 53 | adverts |
| 52 | secondary ed |
| 45 | learn Basic |
| 42 | other languages |
| 41 | primary ed |
| 35 | games listings |
| 27 | competitions |
| 22 | preschool |

Question 10 What other magazines do people get?
105 Micro User   100 Acorn User   52 A&B Computing   26 Beebug
3 Elbug   50 no other magazine.
The table below shows how many of you get how many magazines regularly.

| # mags per person | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|
| # people | 1 | 12 | 30 | 48 | 50 | 50 |
| % of people (approx) | 1 | 6 | 16 | 25 | 26 | 26 |

For over 26% of the User Group, 'Beeblet' is the only source of information devoted to their machine.

Question 11 Who else gets to see your copy of Beeblet?

a) In your household:

| 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| 102 | 52 | 22 | 2 | 1 |

b) outside your household

| 0 | 1 | 2 | 3 | 4 | 6+ |
|---|---|---|---|---|---|
| 152 | 14 | 7 | 2 | 2 | 1 |

plus one school membership of 5 had a readership of 10

<u>Question 12</u> Modems
44 Absolutely interested, 69 within a price range 64 Perhaps 28 Unlikely
Price range where offered was under $300 mostly. Unlikelies also included 2 who already had them, 2 "what's dem?", and 3 not applicable - Electrons!

Question 9 & 13: YOUR OFFERING
If you answered positively, or wanted to, then here's your reminder if you haven't already sent it in.

Feel free to <u>reinvent the wheel</u> if you think you are a beginner - some of us have got some mighty peculiar-shaped wheels to cart our computers round on, and a panel-beating from another learner is always welcome!!

Continued from Page 32

```
32410 .EU LDX #1
              CPX &600
              BNE LVF
              JSR LVS
              LDA #0
              STA S1
              .LU1 JSR LS1
              BEQ LSX
              CMP #97
              BCC LU1
              CMP #123
              BCS LU1
              DEY
              AND #&DF
              STA (S3),Y
              BNE LU1
32420 .ER LDX #1
              CPX &600
              BNE LVF
              JSR LVS
              LDA #0
              STA S1
              LDY S3+2
              LSR S3+2
              .LR1 DEY
              STY S2+1
              LDA (S3),Y
              TAX
              LDY S1
              LDA (S3),Y
              PHA
              TXA
              STA (S3),Y
              PLA
              INY
              STY S1
              LDY S2+1
              STA (S3),Y
              DEC S3+2
              BNE LR1
              RTS
32430         ]:NEXT:ENDPROC
```

# ...SAS...

GCF

I have a host of reasons for being a month behind with the next episode of this Adventure Interpreter, but rather than apportion the blame, I'll simply apologise. I expected to be producing a detailed description of the syntax and activities for this month. There also arises the question of which aspects should be covered within the pages of 'Beeblet', and to what depth; since my working documentation is now up to 11 pages just describing the system.

## ADVENTURE FEATURES

The system as currently designed is capable of supporting the following, with the level of sophistication only limited by your degree of cunning and the inevitable memory constraints (Sideways RAM or 2nd processor anyone?):

a) Magic systems - it is surprisingly easy to incorporate various types of magical systems (eg making an object invisible only requires setting the status flag zero of the object). Several status bits can be used to support sophisticated magic systems, as well as "registers" for holding skill and stamina levels, along with RND to simulate various dice rolls as found in D&D games.

b) Multi-player - SAS can do this, though it wasn't specifically designed to do so. Use SAS to practise setting up solo adventures first.

c) "Virtual"(?) frames - much to my surprise it is possible to simulate an adventure with 'millions' of rooms. One example is an endless tunnel which would allow you to drop objects at any point while still allowing you to get back and see them - this could be implemented within only one frame. More complicated "virtual" mazes could be set up.

d) Fluids - for those who want to handle water realistically. It is possible to allow a bottle of water to be poured into a number of containers rather than just one - ie it need not be either full or empty.

e) Light - all aspects of when and what is visible can be handled, though there are a number of subtleties in the logic for this that required quite a bit of thought.

## DESIGN TRADE-OFFS

A number of facilities could be built into the interpreter to make life easier for the adventure-writer, but they would increase the size and complexity of the program. A careful balance is needed between the size of the DATA and the size of the interpreter. Whilst some facilities are not strictly necessary, they will make life easier for people entering the data statements. Such facilities include representing repeated

phrases as tokens where needed, and defined once only elsewhere.
There is a danger in making the interpreter very sophisticated yet easy to use, that there is no room left for the adventure itself! The fun of writing an adventure game is in making up the scenario, not in spending hours typing any more than is necessary.

CORRESPONDENCE

John H makes some suggestions on machine code implementations:
Recursion: The 6502 already uses a stack for subroutine control, so thal all that is needed is to copy the equivalent of local variables onto a stack at the start of the subroutine, and retrieve them at the end. This could be the processor stack if the amount to be saved is small enough, or it may be necessary to set up another stack for the purpose.
EVAL: This needs more thinking about. I doubt that this function in all its glory is precisely what is required in the system. Probably it would be better to go back to the original requirements and ask what machine code would meet those requirements, rather than attempt a translation from Basic or to emulate the Basic approach. If the full EVAL function is needed, then the obvious approach is to dive into the Basic Rom, where the work has been done already.

Of course, if machine code is being used purely for performance reasons, then it is quite possible to write only a few critical sections in machine code, and have the full power of Basic for everything else. The way Acorn have implemented their assembler makes this a very simple matter to arrange.

Philip Y (Ch'ch) has adopted Aftermath! and Runequest as his preferred games systems, and says ABADS sounds rather like the system he has had in his 'deep thought port' for several years. He suggests moving in the opposite direction from machine code, to Pascal with its wide range of data structures (especially pointers and records). [It would indeed help considerably, but I don't have $$$s for Pascal, nor would everyone else; and as Philip adds, ISO Pascal would need a Random Access file patch as well for from VAS onwards.]
He disagrees with the need for parsed input: an implicit menu for common operations and explicit options for unusual actions would remove the emphasis from having to puzzle out the words expected by the writer to specify a particular action. Unless the parser is very sophisticated and capable of understanding a wide vocabulary, a more standardised input may be desirable, though this may detract from the true 'Adventure' style and flavour. [One way round this is to provide the player with a list of 'understood' words without giving away their meanings].
He proposes also an adventure generation package. [The endless

tunnel above (Virtual frames) is at one end of the scale, and fuller capabilities like this would appear in ABADS].

*** INVITATION TO ADVENTURE WRITERS ***
*** INVITATION TO ADVENTURE PLAYERS ***

WHAT WOULD YOU LIKE DISCUSSED FURTHER?
ARE YOU INTERESTED IN JOINING AN ADVENTURE INTEREST SUB-GROUP?

Please write to Beeblet with your ideas, suggestions and queries - or just an indication of interest.

## SOME THOUGHTS ON MULTI PLAYER GAMES

Steve W.

Having had the privelege lately of playing Multi-user dungeon (MUD for short) via PACNET on the University of Essex Mainframe, it has prompted me to wonder about the possibilities of multi-player games on the Beeb. In the November 1983 issue of Acorn User, there was an article about the mechanics of such a game together with an example (Battleships). All that is required is to set up one BBC (the "host") in such a way that it continually polls (alternately) the keyboard of the host and the serial port (for input from the other BBC, the "terminal"). The terminal would be configured so that everything the user typed in would be SENT out of the serial port, and that everything RECEIVED through the serial port would be VDU'd to the screen. In this way graphics could also be sent as the host machine would set up a trap on the OSWRCH vector and intercept all "screen-write" commands to ensure that the same view appeared on both terminal and host. Alternatively, if each player has a different view on his screen, then terminal and host could both runtheir own versions of the same program and only send data through the serial port whenever one player did anything which could possibly affect the other. I'm surprised that more games aren't available in the UK where modems are so cheap and easily available.

Patrick C. of Lower Hutt has noted some of the new Electron software released in the U.K.:

| | | |
|---|---|---|
| Elite | Acornsoft | (Modes 4,5) |
| Combat Lynx | Durell | (Mode 5) |
| Deathpit | Durell | |
| Mineshaft | Durell | (Mode 5) |
| Bird Strike | Firebird | (Mode 2) |
| The Hacker | Firebird | (Mode 2) |
| Zalaga* | Aarvak | (Mode 5) |

*NB. Completely diffent to BBC version.

# HALL OF FAME

| PUBLISHER: | GAME | HISCORE | SCORER |
|---|---|---|---|
| Aardvark | Frak | 1,026,820 | Andrew P. |
| | Monsters | 175,110 | Geof P. |
| | Zalaga | 215,100 | Peter J. |
| | | 52,090* | Nick C. |
| Acorn | Arcadians | 34,740 | Rasik R. |
| | Aviator | 5,610 | Joanne C's Dad |
| | Countdown | 246 | Andrew |
| | Meteors | 46,200 | Andrew P. |
| | Monsters | 107,390 | Robert T. |
| | Planetoid | 413,875 | Peter J. |
| | Rocket Raid | 140,777 | Mark R. |
| | Snapper | 250,000 | Susan |
| | Starship Command | 9,495 | Robert N. |
| | Super Invader | 59,250 | Kiran R. |
| A&F | Cylon Attack | 157,970 | Aaron |
| | Chuckie Egg | 319,450* | Greg C. |
| Alligat. | Bug Blaster | 37,642 | Steve W. |
| Amcom | Fortress | 89,900 | Andrew P. |
| Bugbyte | Twin Kingdom | 385 | A. Beeston |
| Comp.Con | Android Attack | 236,195 | Peter J. |
| Dr Soft | '747' | 205 | Steve W. |
| Firebird | Bird Strike | 11,340 | David G. |
| | | 14,980* | Patrick C. |
| FG Soft. | Mutant Defender | 6,900 | Steve W. |
| | Lords of Time | 925/1000 | Steve W/Kerry K. |
| | Snowball | 900/1000 | Steve W./Dianne |
| Prog. Pow | Felix in Factory | 13,150 | Duncan's Flatmate |
| | Frenzy | 915,468 | Dianne T. |
| | Killer Gorilla | 91,000 | Susan |
| | Laser Command | 109,225 | P. Leadley |
| | Moon Raider | 112,350 | P.Leadley |
| Sup.Soft | Q*Bert | 43,825 | Peter J. |
| | Mr Wiz | 21,050* | Chris M. |
| | Roadrunner | 118,900 | Duncan's Flatmate |
| ? | Dr Who | 35,240 | Duncan's Flatmate |

*Electron Version. Note: Is there life after Elite?

HALL OF FAME

# USER GROUP MEETINGS

**AUCKLAND:** -meets 2nd Wednesday of the month, 7.30pm, Conference Rooms, Auckland Secondary Teachers College,60 Epsom Ave, Epsom. ph Dave 770-630 x 518(bus) or Kerry 695-355.
-North Shore: meets last Wednesday of the month, 7.30pm, Salvation Army Hall, Glenfield Road, Glenfield ph Chris 419-0543 (wk)

**CHRISTCHURCH:** -fortnightly workshops, Monday 6.30pm at Hagley High School. ph. Michael 582-267.

**DUNEDIN:** -first Thursday of the month. Phone Martin McDowall 42-031

**HAWKES BAY:** -Alternative months Hastings & Napier. Contact Kendall Napier 435-624, Bob Taradale 446-955, Mitch Hastings 778-235.

**INVERCARGILL:** -Phone Evan 330-444 Invercargill.

**PALMERSTON NORTH:**-Phone Tom Skinner 83-009(wk)

**TAUPO:** -meets every second Wednesday @ 7pm. Contact David 84-215 (h), 85-124 (wk).

**TAURANGA:** -every second Wednesday. ph. Chris & Jane 65-076.

**TIMARU:** -Phone Lloyd van der Krogt Timaru 61-412.

**TOKOROA:** -meets first Friday of each month, Tokoroa High School, 8pm. Phone Brian 67-625 Tokoroa.

**WAIKATO:** -first Friday of the month 7.30pm Waikato Tech Institute. Ph. Alison or John, Morrinsville 6695.

**WELLINGTON:** -meets last WEDNESDAY of the month, 7.30pm, Royal Society Lecture Theatre, Turnbull St, Thorndon.Workshops: second Thursday of month, other venues. ph. Don 848-235 or Beryl 862-874 or Mike 785-437.

**OTHER CENTRES:** - let us know and we will publish details here.

## THE NEWSLETTER:

**CONTRIBUTIONS**:-most welcome. Listings should be sent in on tape or disc. (Tape - 2 copies, one at 300 baud please). Please include written explanation of listing; text and graphics only material welcome too.

**DEADLINES**:- For a particular issue, the last day for material is the last day of the month prior to publication.

**ADVERTISING**:- Rates are $20 per half page (camera ready), deadline as above.

Copyright (c) **BBC/ACORN COMPUTER USER GROUP OF NZ INC.**

Published monthly except January and mailed to financial members.

## MEMBERSHIP:

Membership of the users group is on payment of an annual subscription from April to March. For the 1985/6 year it is: $30.00, plus $5.00 for Auckland & Wellington members as a local branch levy, which entitles attendance at meetings as detailed above. Joining or renewing members get all the back issues of the current year - those joining from 1 October may have half the annual sub waived and only receive newsletters from 1 October.

THE BBC/ACORN COMPUTER USER GROUP OF NZ INC, P.O.Box 9592, WELLINGTON.

Reproduced by ROBERTS PRINTING CO. LTD, WELLINGTON.